DOCUMENT RESUME

ED 084 839 EM 011 660

AUTHOR Galler, B. A.; And Others
TITLE CRISP: An Interactive Student Registration System.
INSTITUTION Michigan Univ., Ann Arbor.
PUB DATE 24 Jan 73
NOTE 27p.; Paper presented at the ACM Annual Conference (Atlanta, Georgia, August 1973)

EDRS PRICE MF-$0.65 HC-$3.29
DESCRIPTORS Class Activities; Computer Oriented Programs; Computer Programs; Computer Science; *Computer Science Education; Data Processing; Educational Administration; Electronic Data Processing; Higher Education; *On Line Systems; Program Descriptions; Programing; *Student Projects; Systems Approach; *Systems Development
IDENTIFIERS *Computer Registr Involving Student Participation; Data Systems Center; Interfaces; Michigan Terminal System; Project CRISP

ABSTRACT

A class project in a systems programing course at the University of Michigan sought to produce a system realistic enough to warrant production use, the underlying assumption being that the reality of the project would motivate students and provide them with valuable experience. Eighteen experienced students in the Computerized Registration Involving Student Participation (CRISP) project worked with the Michigan Terminal System to produce an on-line course registration system for students which would interface with those main parts of the current system dealing with room scheduling, transcript production, etc. Interfaces with the current Data Systems Center were ordered, student groups with individual functions were organized, and internal and external specifications stipulated. The student-designed system was sucessfully demonstrated, resulting in support from the cnetral university administration for further development and eventual implementation of the system. The conclusion was therefore reached that a class project could produce a realistic system, 1) provided that good tools are available, particulary a general-purpose time-sharing system, and 2) that continual guidance on the interface with the system's eventual environment is maintained. (Pb)

FILMED FROM BEST AVAILABLE COPY

ED 084839

CRISP: An Interactive Student Registration System

B.A. Galler, R. Wagman, J. Bravatto, G. Lift,
G. Kern, V. Berstis, and E. Munn

The University of Michigan, Ann Arbor, Michigan

January 24, 1973

U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

EM 011 660

Index Terms

Interactive, On-Line, Programming, Systems Programming, Student, Administrative System

Abstract

A class project, even for an advanced class in systems programming, is usually simplified to make it possible to achieve success. A conflicting goal is to produce a realistic enough system to warrant production use, so as to provide the class with sufficient motivation and the right kind of system experience. The problem is compounded when the resulting system is to be a component in an on-going information-processing system and must be compatible with it - with existing documents and procedures, with unstated assumptions about the data and about the current algorithms used to process the data, and with the social and institutional traditions that surround it. This paper reports on one successful class project of this kind.

CRISP: An Interactive Student Registration System

B. A. Galler, R. Wagman, J. Bravatto, G. Lift,

G. Kern, V. Berstis, and E. Munn

Introduction

In 1971, Arden, Flanigan, and Galler [1] reported on a systems programming class based on a group project in some area of systems. Clearly, with 15-20 students, over a period of a semester, one can undertake a project of some magnitude, but when one considers the need for specifying the system to be produced, and the programming and debugging time involved, there is always pressure to simplify the system goals. After all, one of the desired outcomes is the successful completion of the project, after facing up to the many difficulties involved.

On the other hand, if the students were to become convinced that the exercise is "purely academic," there is a problem in motivation and dedication. Students are always involved in a number of other activities besides the particular course in which the professor is so involved, and if their morale and dedication is weakened, the system project will be endangered. The difficulty, therefore, is to find a problem that is challenging, educational, and feasible, while at the same time realistic and with enough promise of generating a production system to motivate the students to put a great deal of time and effort into it. It is also important, of course, to choose very good students with sufficient

programming background that they can plunge directly into the problem with a minimum of introductory material, and to provide them with good tools. In this case, the 18 students (primarily graduate students) were all quite good programmers, some with one or more years of experience on campus research projects, and they had available to them the facilities of the Michigan Terminal System (MTS), an excellent, general-purpose time-sharing system [2].

The Problem

During the months preceding the course, the university administration had been considering the adoption of an alternative to the current system of assigning students to classes. On a campus in which administrative and academic decision-making is highly decentralized, the system currently in use reflects the historical pattern of growth in which small changes are superimposed on each other in response to demands from many different sources. Changes are difficult to implement, and their consequences difficult to predict. Although the administrative Data Systems Center (DSC) is moving rapidly toward efficient on-line access, this was not yet being considered as a viable alternative for the student registration problem. This, then, was an excellent problem for the class; viz, produce an on-line "reservation system" for students, which would interface with the main part of the current system; e.g., the part concerned with room scheduling, transcript producing, time schedule printing, etc., and which would be capable of handling the volume presented by 35,000 students taking courses in any or all of the Fall, Winter, and Spring-Summer terms, and the Spring and Summer half-terms. Moreover, the current lines of students

waiting to register should disappear, and the whole thing should cost relatively little to run! Finally, while the class-generated part of the system would be developed to run under MTS, it would have to be planned so that eventually it could be moved to the DSC computer.

The Interfaces

One immediate problem was the specification of the interface with the current system. Since neither the professor nor any of the students who could be expected to enroll would be very familiar with the current system at DSC, representatives from DSC were solicited to join the class as observers. They would advise on the current system's organization and insure compatibility at the interface. (One very competent person from DSC subsequently enrolled in the course, proving invaluable for these interface requirements.) One of the authors of this paper joined the class as a "participating observer" (no casual observers were allowed, because of group morale considerations), and provided very welcome and useful insights into the counselling interface that the system would have. One additional interface was underrepresented; e.g., the Office of the Registrar, where many of the policy decisions were being made, and while the difficulties thus introduced have now been largely overcome, it points up a danger in such an undertaking. In particular, in areas where the current system is ambiguous or unclear, assumptions were made during the course which might have been different if the Office of the Registrar had been consulted at the time. Fortunately, the final CRISP* system was

*CRISP: Computerized Registration Involving Student Participation

flexible enough to accommodate most of the changes they wanted to make.

The charge to the class on the first day is included here as Appendix A. The interface to the present system was to be as follows: Tapes would be brought from DSC with the current student information file, the master catalog of courses, and the specific offerings for each term under consideration (up to five at any time). These would be used and maintained by CRISP in MTS, and subsequently, after students were assigned to courses and sections, tapes would be generated for further processing by DSC.

The Class

The organization of the course was essentially that described in [1]. The class was divided into groups, as suggested in Appendix A, although on the first day, the students proposed combining some of the suggested groups and reassigning some of the responsibilities. As a result, the following groups were formed:

Group 1 - Data base initialization programs and batch output program specifications.

Group 2 - Interactive program to accept input from terminal operator, check for syntactic validity, and call on group 3 routines to interact with data base. Generation of user's manual.

Group 3 - Program to call up student record and appropriate course records and process commands as provided by group 2.

Group 5 - Manipulation and maintenance of student and course data bases.

Group 7 - Interrupt handling and post-processing cleanup of data bases via batch maintenance program.

Group 8 - Project management and coordination.

Each student was asked to specify his own choice of group, but he was requested to avoid any group for which he already had experience and expertise. (Here the educational goals of the course certainly conflicted with the production goals.) Then the expected problems of social organization began to appear, in terms of internal group leadership, a request for transfer of a dissident from one group to another group which considered itself large enough already, and communications interfaces between groups.

The project management function was the responsibility of one of the groups. They tried several devices to facilitate intergroup communication, such as flow analysis, group progress reports, and written specifications for interfaces, and some of these were quite successful. Others, however, were abandoned as too time-consuming for the benefits. The professor attended all classes (and some extra-class group meetings), and offered advice, but remained largely as a monitor and "friend of the court," as well as a buffer to the rest of the university. In some cases, the students ignored this friendly advice - and learned from it. An example of this was the suggestion that in writing code, each group should use distinctive symbols suggestive of the group designation. They considered this unnecessary, but the subsequent confusion over multiply-defined symbols required a fair amount of extra effort.

Specification

As seen in Appendix A, the system needed not only internal specification, but external specification as well.* Because of the need for compatibility with the current system, and credibility

*Some minor features mentioned in Appendix A were eliminated by the class.

with those familiar with current procedures, it was quickly decided that terminal operators using CRISP would extract the information they needed for input from existing documents. But the interactive responses to the input, error diagnostics, and confirmation of results, as well as the necessary on-line file maintenance commands and interaction, had to be specified. Our "participating observer" offered to produce a user's manual, which would represent the external specifications, and this offer was enthusiastically accepted by all. The class wisely counselled the group concerned with the interactive part of CRISP to organize that area into tabular constructs to facilitate changes. In fact, this was subsequently proven to be very wise, in the ability to add and modify many commands and access authorization codes relatively easily. It was largely in this area that it proved possible to accommodate the wishes of the Office of the Registrar after the course was over.

Implementation

One lesson the group learned was that the rather enjoyable period of arguing over specifications (both external and internal), must come to an end eventually, and hard decisions must replace wishful generalizations. Data structures must be determined, good estimates of expected volume of transactions must be obtained, and subroutine parameters must be agreed upon; and then a very tight rein must be kept on any further revisions. In this case, the specification phase probably lasted two or three weeks too long. It was clear to everyone that this was happening, but the feeling persisted that the specifications were not yet tight enough;

that they had not really faced up to some of the real-world constraints that CRISP would have to live with. As time ran out, some design compromises were finally made, and coding began.

It became apparent that not very much time remained for coding and debugging, so a practice was initiated of having each group report its progress to the class at each meeting, in terms of percentage of expected code already written, and percentage thought to be debugged (admittedly difficult to determine!). This peer pressure for not delaying the group project was very effective, but integration of the various components produced by the several groups (and only partially debugged in many cases) did not start until the last week of the term.

Here the debugging and other on-line facilities of MTS proved invaluable. During the last week of the course, the system was debugged enough to allow a demonstration two weeks later before a number of representatives of the administration. (Of course, only those parts of CRISP which could be relied upon were invoked during the on-line presentation, but the kind of typical activity shown in Appendix B was possible even then.) The debugging sessions were held at a terminal near the batch facility of the Computing Center, so large listings and other printouts could be generated very easily. One feature of the MTS system which turned out to be particularly useful (besides the interactive symbolic debugging system and the system on-line editor), was the ability to suspend activity on one "logical telephone line" and request access to the MTS system via another "logical line," using the same physical line. On this second line one could sign on to MTS under another

identification number* and modify or permit files to be accessed by the original task. This kind of flexibility is very useful when integrating many components developed in files belonging to different groups working under various identification numbers.

Another development in the MTS system, which was anticipated during the course, but which was not actually available until four months later, was a revision in the MTS file management system to authorize a number of terminals to read and/or write a common set of files. Such a facility is clearly needed for an on-line reservation system, and with the external specifications already available during the course, the students were able to include multiple terminals in the design.

Subsequent Developments

As a result of the administrative demonstration, a small amount of money and some computer time were authorized to support a subset of the students during the following summer to continue the debugging of CRISP. A committee of representatives of the Office of the Registrar, the Counselling Office of the major undergraduate college, the CRISP class, DSC, the Scheduling Office, and students and faculty from other schools and colleges within the university, began to meet throughout the summer to plan a pilot run of CRISP, using real data. The CRISP group encouraged the use of actual student elections, but advocated a "dry run," so any problems which might develop would not jeopardize the credibility of the system

*More recently, under the same number as well.

with the campus community. The pilot run was carried out in September, using about 12% of the actual advanced classification student elections for the Fall term. There were a few small problems, but the results were very gratifying. At the time of writing, the committee is currently preparing a recommendation on the use of CRISP for the university.

One interesting question which still remains is whether to continue to use CRISP on the MTS system in conjunction with the DSC system, or to convert CRISP to run entirely on the DSC computer. In either case, there is the transitional problem of communicating what is known about CRISP and its internal structures to the staff of DSC for continued maintenance and development. With all of their good intentions, only a few students had the necessary time available for adequate documentation. It is clearly an integral part of system design, but the magnitude of the project forced this aspect to be less than satisfactory. Of course, system documentation was eventually generated, but the lesson was learned by all concerned.

Another lesson was involved with the ease with which the CRISP system could be moved entirely to the DSC computer system. Although this had been a goal from the start, and indeed the code had been made modular for this purpose, a number of system dependencies (i.e., implicit assumptions about the use of MTS) did creep in. For example, since each terminal runs an independent CRISP task on an independent copy of MTS (while sharing the actual re-entrant code of MTS), the CRISP system did not itself have to worry about interacting with more than one terminal. This would require very different treatment in the IBM Operating System at DSC.

An interesting byproduct of the CRISP work was an interactive room scheduling and reservation system developed by one of the authors for use in the Scheduling Office. When the possibility was raised that the use of CRISP might indicate classes that would overflow assigned rooms, or that additional sections of a popular course would be needed, but academic departments might not be able to obtain new classrooms quickly enough, an interactive on-line reservation system was suggested. After a simple model was developed and implemented by Professor William Riddle with his beginning programming class during the summer, more realistic specifications were written, and the system was implemented and delivered to the Scheduling Office.

Conclusions

It is indeed possible to take on a realistic system with an advanced class, provided a good set of tools is available; in particular, a flexible general-purpose time-sharing system. Some prospect of eventual adoption of the results, if warranted, can provide a great deal of motivation. On the other hand, care must be taken to have continual guidance on the interface with the eventual environment in which the system will be used.

The professor must himself be enthusiastic about the project--student criticisms of the course emphasized this point - but he must be sure to provide a careful balance between student management of the project and his "friendly advice." He must also provide enough initial specification to start the design process, but not so much as to stifle student initiative. The real challenge is to

encourage the students in their natural enthusiasm and creativity, temperéd with an appropriate sense of the real world.

Appendix A:

Proposal for CCS 673 Project CRISP* - B. A. Galler

1. General description - The problem is to develop a prototype (hopefully capable of expansion to a full system for the University of Michigan) for the student registration process, excluding payment of fees, etc. The general operation of the system will be as follows: A student will consult a printed Time Schedule as at present, including consultation with his counselor, and taking into account information on closed sections. He will present a completed course election form to one of several terminal operators, who will display any elections already confirmed for that student, and then key in his elections and request space in his selected courses. The system will confirm his elections if possible, and the student will receive written confirmation at a later time. After confirming a student's current elections, the program will request an optional list of preferences for the following term, taken from a list of courses to be offered. Students will be given an opportunity to indicate desired, but not listed, courses as well. Various kinds of queries will be allowed during the process, as well as changes to the Time Schedule information which forms the initial data base for the system.

2. Input - The initial data base will consist of a selected portion (capable of expansion) of the standard Time Schedule, together with coded information as to whether certain combinations must be elected together (such as lecture and lab), and whether certain

---

*Computerized Registration In Spite of Problems

courses are cross-listed (so enrollment quotas must be combined), etc. In addition, enrollment quotas and "early warning" levels should be entered, if appropriate. (If possible, enrollment quotas should be maintained by major unit, school or college, by grad/undergrad, by class year, and by concentration. This may be considered a special case of a general need for special restrictions to be invoked for individual courses, except that it would be desirable to be able to get at the quotas dynamically to change them.) Provision should be made for a course to be designated as having a waiting list once any enrollment quota has been reached. (If the quota is later raised, places will be filled first from the waiting list according to order of arrival. Thus, if an instructor wishes to reserve to himself the option of selecting a subset of the people who wish to register for his course, he can specify a quota of zero, but request a waiting list.)

3. Terminal Operation - The system should support several kinds of terminals, including a display device (such as the CDC terminal), a teletype or typewriter device, and the touch-tone telephone ARU device. Selective responses should be available which are appropriate to the nature of the device. (Input should normally be echoed, with additional information added for verification purposes, such as course title and/or time at which class meets.) If the requested schedule cannot be confirmed, as many reasons should be given as possible, so as to minimize the number of return trips necessary to obtain an acceptable schedule. (An attempt to register for more than one course at the same hour should be flagged with a warning, but should not be treated as a "fatal error.") If a

request for space in a closed course is made, it should be possible to request at the same time (or else be prompted) that the student's name be added to the waiting list. Provision should be made for periodic listing of closed courses for posting, and for interrogation by touch-tone telephone (and spoken response from the ARU) by any student as to the status of any course. The report of the status should give number enrolled and number of available spaces or length of the waiting list. (One might expect a few such telephones to be connected throughout the day for general student use.) Provision should also be made for dropping or adding individual courses by students who have already registered.

One program (with restricted access) should be designated the "master program," and this should be (the only one) authorized to change every part of the Time Schedule data, including quotas. Courses or sections may be deleted here, with lists of those already registered for them available. Other programs will be allowed to change quotas for a specific department's courses. Included here is the ability to set or change "early warning" levels, for flagging potentially closed courses. Still other programs, for general use, should be able to query the status of various parts of the data base, such as the number enrolled under any quotas specified, the length of the waiting list, whether a particular section is closed, how many students have enrolled so far, how many today, a particular student's elections already confirmed, etc.

4. Output - The system should supply on demand a printed report of the current status of all (or a subset of) course enrollments, a closed-course list (including a separate list of changes -

additions and deletions - since the last closed-course list appeared), a list of students enrolled in each course (with an additional list of names on the waiting list), and a schedule for each student showing his confirmed elections.

5. Data Base Management - The new file sharing features of MTS should be available in time for this course. Provision will have to be made for back-up and recovery in case of system crashes, as well as temporary lock-out on elections until they are confirmed or rejected (including delays due to prompting for waiting-list action).

6. Documentation - In addition to documentation of the internal organization of all components of the system, there should be a User's Guide which would include instructions for preparation of input data, instructions for operating a terminal, instructions to a student as to how to specify his desired elections and query for closed courses or the status of a particular course, and a list of possible diagnostics and the causes for their occurrence.

7. Suggested Functional groups for class implementation of CRISP -

Function	Approximate Size of Group
1. Input Specifications, Initialization of data base, Specification of Storage Structures	3
2. Command Language Specifications and Interpreter	3
3. Terminal support and Query Programs	2
4. Special Course Restriction Routines, System Macros	2
5. Data base management, protection, interlocks	3
6. Generation of output	2

7.	Restart, recovery, back-up	2
8.	System integration, maintenance of system workbook, project coordination, code efficiency	2
9.	User's guide and test environment, interaction with Registrar's Office	2
	Total	21

8. Specific Responsibilities of Functional Groups -

8.1 Input Specifications, Initialization of Data Base, etc. - This group will obtain input tape from Administrative Systems, modify it as necessary to include quota information, early warning levels, relationships between lectures and labs, etc., and indications of courses which require special "restriction subroutines." This group is also responsible for specification of the storage structures to be used and for routines for bringing this data into appropriate files. A similiar responsibility exists for the list of courses for the following term(s) for preference tabulation. Routines should be provided for updating these tapes as necessary after acquisition from Administrative Systems.

8.2 Command Language Specifications and Interpreter - This group specifies the various commands for interrogating the data base, entering student elections, and modifying the data base. They will provide appropriate prompting routines, and an interpreter which will analyze commands and issue calls on various routines as necessary. (The called routines will be generated by other groups.)

8.3 Terminal Support and Query Programs - This group will determine the appropriate forms in which input/output is to occur on the various devices to be used. (Some prompting or display might be appropriate for one type of device and not another.) This group will also implement the routines which search the data base and which modify it. (These routines will call upon primitive routines provided by group 8.5.)

8.4 Special Course Restriction Routines, System Macros - This group will generate the special course "restriction subroutines" needed to enforce particular rules. (This is not the checking of prerequisites, which is not intended here.) These subroutines will be invoked during the attempt to confirm a student's elections. This group will also maintain the file of macros developed by the course activity.

8.5 Data Base Management, Protection, Interlocks - This group will provide the primitive routines used by the query and modification programs in searching and changing the data base. This group will provide appropriate checks for authorization, interlocking, obtaining and freeing space, etc.

8.6 Generation of Output - This group will provide routines for printed reporting of confirmed student elections, class lists, waiting lists, deleted course lists, closed course lists, etc. These routines will be invoked via the Command Language Interpreter. They will also generate a final status tape to return to Administrative Systems for later processing.

8.7 Restart, Recovery, Back-Up - This group will advise the others on appropriate code to be used to facilitate recovery and restart after system failures. Routines will be provided for back-up as necessary. Restart and recovery procedures will be specified and implemented.

8.8 System Integration, Maintenance of System Workbook, Project Coordination, Code Efficiency - This group will assume coordination responsibility, setting goals and monitoring them. Approval of changes to project specifications resides here, as well as monitoring individual groups' documentation and code efficiency. The system workbook will be maintained in one or more files by this group, and they will implement a skeleton system for use during system integration.

8.9 User's Guide, Test Environment, Interaction with Registrar's Office - This group will interact with and represent the user community for this system. They will lobby for whatever improvements are deemed desirable, and they will be aware of the user interface. They will generate a User's Guide suitable to instruct terminal operators and students to interact with the system and to interpret the output to be produced. They will create a test environment for the system.

MODIFICATION 1 - Conditional Enrollment Threshold (CET)

In the CRISP proposal, provision is made for "early warning" levels, which would initiate messages to departments when enrollment reached preset levels. We now replace the "early warning" concept with the Conditional Enrollment Threshold (CET), which will

be set (and dynamically changed) by departments, just as quotas are set and changed.

When the enrollment in a section has reached the CET level, requests for enrollment by students will be rejected unless a particular "counselor priority" indication is entered for that course. The operator will enter it only when a signed message is received from a counselor. If the enrollment in a section has in fact reached the section quota, "counselor priority" requests will be queued in the order of arrival at the head of the waiting list, so that such people will be the first admitted to a course if the quota is subsequently raised.

The CET is intended to reserve places in sections for those whom counselors are willing to indicate as having special problems and who need special priority. A department may set the CET to zero, requiring special permission for everyone, or they may set the CET equal to the quota for a section (the default case), in which case no counselor priority would be needed for anyone. The response to an interrogation about enrollment in a course should indicate whether the CET has been reached. If there is a waiting list, the length of the CET portion and the length of the non-CET portion would be reported, also.

MODIFICATION 2 - Piecewise Registration

It is quite possible that registration procedures will be spread over several weeks, and that different groups of students will be registered in different time periods, such as by alphabetical groupings. In this case, it would be desirable to increase the enrollment quotas whenever a new time period began, without

necessarily admitting those on the waiting list from the preceding time periods. The program which allows departments to change quotas, therefore, should allow for modification of quotas without the automatic entry into the course of people from the waiting list.

Appendix B - CRISP Output

```
  YOU ARE NOW CONVERSING WITH CRISP
*id 2145273192

  214-52-7319-2 WIRE BARBARA                    77 L
  10:04.18  01-29-73
  *ELECTIONS FOR FALL   *
*el

  *ELECTIONS FOR FALL   *
   1 353 C C S     473   3 001
   2 353 C C S     673   3 001
  TOTAL CREDIT HOURS FALL     6
*id 1234567899
  ** NO RECORD:  123-45-6789-9
*insert 1234567899 name=wilson pickett fence unit=1 year=68

  INSERT 123-45-6789-9 WILSON PICKETT FENCE      68 L
?ok
  DONE  10:05.43
*st 353 673 001

     FALL 353 C C S     673 SYSTEMS PROGRAMMING
  SEC       LIMITS      ENROLLED    AVAILABLE        WAITING
  001  MAX 21  CET 10    TOT 1   REG 9  PRI 11   REG 0   PRI 0
*id 1234567899

  123-45-6789-9 WILSON PICKETT FENCE      68 L
  10:07.31  01-29-73
  *NO ELECTIONS FOR FALL   *
*ad 353 673 3 001
*ad 428 555 3 001
*ad 428 684 3 001
*hr 9
  ECHO:
   1 ADD    353 C C S     673  3 001
   2 ADD    428 MATH      555  3 001
   3 ADD    428 MATH      684  3 001
  TOTAL CREDIT HOURS FALL      9
*update

     FALL ELECTIONS CONFIRMED FOR  9 HOURS CREDIT 10:08.51

   *ELECTIONS FOR FALL   *
   1 353 C C S     673  3 001
   2 428 MATH      555  3 001
   3 428 MATH      684  3 001
  TOTAL CREDIT HOURS FALL      9

     D:0     A:3     W:0     N1:1     N2:0   K28D

  KEEP THIS, IT'S NOT LITTER
```

```
*id 2145273192

  214-52-7319-2 WIRE BARBARA                    77 L
  10:09.39  01-29-73
  *ELECTIONS FOR FALL    *
*drop 353 473 3 001
*ad 428 555 3 002
*hr 3
  ECHO:
   1 DROP    353 C C S     473  3 001
   2 ADD     428 MATH      555  3 002
  TOTAL CREDIT HOURS FALL       3
*up

     FALL ELECTIONS CONFIRMED FOR  6 HOURS CREDIT 10:11.02

  ** CHECK HOURS! ENTERED:  3  CONFIRMED:  6

  *ELECTIONS FOR FALL    *
   1 353 C C S     673  3 001
   2 428 MATH      555  3 002
  TOTAL CREDIT HOURS FALL      6

    D:1     A:4     W:0     N1:1     N2:1   K28D

  GUARD THIS WITH YOUR LIFE (IT IS YOUR LIFE)
```

```
*di 353 673 3 001
  ** INVALID SECTION NUMBER:  3
*di 353 673 001

     FALL 353 C C S     673 SYSTEMS PROGRAMMING
  SEC        LIMITS     ENROLLED     AVAILABLE          WAITING                    HRS
  001  MAX 21  CET 10    TOT 2    REG 8  PRI 11    REG 0   PRI 0      a LEC   3  TTHS   8PM          2086 F B
*di 428 555 000-999

     FALL 428 MATH      555 COMPLEX VARIABLES
  SEC        LIMITS     ENROLLED     AVAILABLE          WAITING                    HRS
  001  MAX 40  CET 10    TOT 1    REG 9  PRI 30    REG 0   PRI 0      a LEC  1-3 MWF    10           433 P A
  002      40      10        1        9       30        0       0      a LEC  1-3 MWF    1PM         3033 E E
*ca 338506782
  338-50-6782-4
*in 3385067824 name=jim year=72 u=r

  INSERT 338-50-6782-4 JIM                          72 R
?ok
  DONE  10:15.07
*in 2145273192 a_
*in 2145273192 na=this is a duplicate zilch u=e y=73

  INSERT 214-52-7319-2 THIS IS A DUPLICATE ZILCH 73 E
?ok

  ** ALREADY ON FILE - 214-52-7319-2 WIRE BARBARA                77 L
*cc 353 673 001 max=15 cet=5

  CHANGE FOR FALL    353 C C S    673 001 SYSTEMS PROGRAMMING
   MAX  CET              HRS
    15    5      a LEC   3  TTHS   8PM          2086 F B
?ok
  DONE  10:17.35
*di 353 673 001

     FALL 353 C C S     673 SYSTEMS PROGRAMMING
  SEC        LIMITS     ENROLLED     AVAILABLE          WAITING                    HRS
  001  MAX 15  CET 5     TOT 2    REG 3  PRI 10    REG 0   PRI 0      a LEC   3  TTHS   8PM          2086 F B
*qi
  NO ACTIVE ID
*qt
  TERM  FALL
*help

  THE FOLLOWING COMMANDS WILL BE ACCEPTED:
  ID TE IN CH RE ST DI LC AC CC SE DC CA XC HE CO MT SY SI
*sy
#EXECUTION TERMINATED
```

References

Arden, B. W., Flanigan, L. K., and Galler B. A. *An Advanced System Programming Course*, Ljubljana, Yugoslavia: Proceedings Congress of Int'l Federation of Info. Proc. Soc., Vol. TA-7, pp. 115-119, 1971.

Alexander, M. T. *Organization and Features of the Michigan Terminal System*, Montvale, NJ: AFIPS 1972 SJCC Proceedings, pp. 585-591, 1972.